INTERNATIONAL JOURNAL FOR NUMERICAL AND ANALYTICAL METHODS IN GEOMECHANICS, VOL. 21, 825–843 (1997)

A NUMERICAL REPRESENTATION OF FRACTURING GRANULAR MATERIALS

D. ROBERTSON*, M. D. BOLTON† AND G. R. MCDOWELL‡

Department of Engineering, University of Cambridge, Cambridge CB2 1PZ, U.K.

SUMMARY

This paper describes the computer algorithms used in a numerical simulation of the compression of an aggregate of crushable grains. It has been used in a model[1] for the evolution of a granular medium under one-dimensional compression, in which the probability of fracture for individual particles is a function of applied stress, particle-size and co-ordination number. The information relating to the particles is represented in a compact way on the computer which allows the number of particles produced to become sufficiently large for satisfactory comparisons to be made with experimental data and which allows information, such as the positions and sizes of the particles, to be easily extracted. An algorithm based on the representation is used to locate neighbouring particles in a way which does not deteriorate unacceptably in terms of speed as the number of particles increases. © 1997 by John Wiley & Sons, Ltd.

Int. J. Numer. Anal. Meth. Geomech., Vol. 21, 825–843 (1997)
(No. of Figures: 13 No. of Tables: 0 No. of Refs: 8)

Key words: fracture modelling; granular materials; numerical modelling

INTRODUCTION

The Fractal Crushing of Granular Material[1] introduces a two-dimensional numerical model in which a granular medium under stress evolves as a function of the behaviour of individual grains.

The aim is to model fragmentation in discrete elements. Other discrete element methods[2] are concerned with problems of elastic/sliding contacts of invulnerable discs or balls. The focus here will be to model fracture in the absence of any explicit sliding or elasticity; these additional features will be included in due course.

Previous workers have pointed to the fractal nature of a debris of crushed fragment.[3–5] Bolton and McDowell[6] advance the hypothesis that both sands and clays adopt such a fractal geometry on their 'normal consolidation line'. A better understanding of the successive fracture of intact rocks, or of an aggregate of particles, is called for.

The numerical modelling involved has to face the difficulty of tracking the existence and co-ordination of breaking fragments. The main objective in this paper is to establish efficient algorithms for this purpose.

*Correspondence to D. Robertson, Research Student in Geotechnical Engineering, Department of Engineering, University of Cambridge, Cambridge, CB2 1PZ, U.K.
†Lecture in Engineering
‡Research Student in Geotechnical Engineering

CCC 0363–9061/97/120825–19$17.50

Received 8 August 1996
Revised 9 April 1997

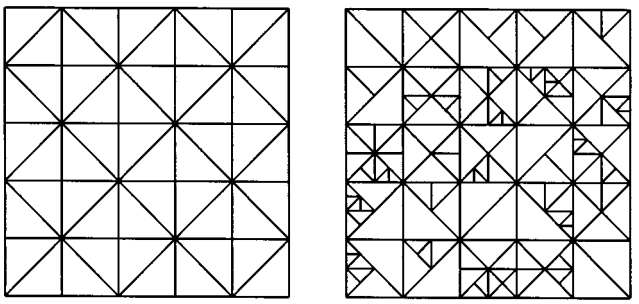

Figure 1. Initial grid of triangles and model after several iterations

The model represents an initial sample comprising uniformly sized grains by an array of 50 identical right-angled triangles (Figure 1). As a macroscopic stress is applied, and gradually increased, each triangle may fracture to create two right-angled triangles in its place, and so on. At each iteration, triangles fracture with a probability of $1 - P_s(d)$ where $P_s(d)$ is the survival probability for a triangle for which the lengths of the two equal sides are d. In the present simulation $P_s(d)$ is given by

$$P_s(d) = \exp\left\{-\left(\frac{d}{d_0}\right)^2 \frac{(\bar{\sigma}/\sigma_0)^m}{(C-2)^a}\right\} \tag{1}$$

where d_0 is the size of the original triangles and C is the co-ordination number (number of neighbours), $\bar{\sigma}$ is the applied macroscopic stress and, for normalisation of the plots, the tensile strength σ_0 of the initial triangles is set at unity. The grid of triangles shown in Figure 1 is regarded as being embedded in a larger region of similar triangular elements, so that the initial co-ordination numer C of every triangle, including those at the edges of the sample, is 3.

Equation (1) is based on the Weibull statistics of brittle fracture. Weibull recognises that the survival probability of a block depends on the survival of all its constituent parts: consequently, the survival probability is greater for smaller specimens, since they contain smaller and fewer flaws. For a specimen, volume V, under tensile stress σ the survival probability is given by Weibull as

$$P_s(V) = \exp\left\{-\left(\frac{V}{V_0}\right)\left(\frac{\sigma}{\sigma_0}\right)^m\right\} \tag{2}$$

where σ_0 is the tensile stress which causes 63 per cent of samples to fail.

For a real, 3-D spherical particle, loaded diametrically between flat plattens (i.e. with a minimum possible co-ordination), the characteristic tensile stress induced in the particle can be defined as

$$\sigma = \frac{F}{d^2} \tag{3}$$

following Lee[7] and Jaeger[8]. The survival probability of the particle is then

$$P_s(d) = \exp\left\{-\left(\frac{d}{d_0}\right)^3 \left(\frac{\sigma}{\sigma_0}\right)^m\right\} \tag{4}$$

McDowell *et al.*[1] showed (4) to be consistent with Lee's[7] data for the tensile strength of rock grains compressed under diametrical forces.

For a regular cubical array of identical spheres, each of diameter d, under uniaxial compressive stress $\bar{\sigma}$, the effective load on each particle would be $\bar{\sigma}d^2$. This would result in an induced tensile stress equal to the macroscopic compressive stress on the aggregate $\sigma = \bar{\sigma}$, so that the survival probability of grains is given by (4) with $\sigma = \bar{\sigma}$.

For the 2-D simulation, it is evident that (1) mimics (4) for particles with a minimum co-ordination of $C = 3$. Jaeger[8] showed that the tensile stress induced in a particle reduced as C increases, since the forces on its boundaries are spread out over many contact points. This is modelled by the equation:

$$\sigma = \frac{\bar{\sigma}}{(C-2)^{\alpha}} \tag{5}$$

where the factor α can be used to vary the significance of the co-ordination number on the induced tensile stress. A value of $\alpha = 0$ would imply that the induced tensile stress is independent of the co-ordination. In this case, the largest particles would always be the most susceptible to breakage under increasing macroscopic stress, leading to a uniform array of fine grains. However, it will be shown that when the effect of the co-ordination number dominates over particle size in determining the fracture probability of a grain, the model predicts the evolution of a distribution of particle sizes, such that some of the largest particles remain intact under the protection of many neighbours, in agreement with available data.

In order to explore the onset of fractures at low values of $\bar{\sigma}$, an initial value of $\bar{\sigma}$ is chosen such that only one of the initial 50 triangles would be expected to break. This would imply a survival probability of 0·98 for each of the initial triangles.

For example, taking $d = d_0$ and $C = 3$ initially:

$$\bar{\sigma} = \left\{\ln\left(\frac{1}{0{\cdot}98}\right)\right\}^{1/m} = 0{\cdot}458228 \text{ for } m = 5$$

The increment operator for stress is taken so that the increment at each iteration is small and so that the increment in $\ln \bar{\sigma}$ is constant:

$$\bar{\sigma} = (1{\cdot}01)\bar{\sigma}$$

Figure 1 is taken to represent a logical map of the neighbourhood relationships of particles. In a granular medium there will obviously be voids, which are not explicitly represented. However, the work done per unit volume by the macroscopic stress $\bar{\sigma}$ can equated to the work absorbed in the fracture and frictional rearrangement of grains. For this purpose, McDowell *et al.*[1] introduced the following work equation:

$$\bar{\sigma}\,\mathrm{d}\bar{\varepsilon}^{\mathrm{p}} = \mu\bar{\sigma}\,\mathrm{d}\bar{\varepsilon}^{\mathrm{p}} + \frac{\Gamma\,\mathrm{d}S}{V_{\mathrm{s}}(1+e)} \tag{6}$$

The left-hand side of (6) gives the plastic work done by a uniaxial stress $\bar{\sigma}$ for an irrecoverable increment $\mathrm{d}\bar{\varepsilon}^{\mathrm{p}}$ in uniaxial strain $\bar{\varepsilon}$. The first term on the right-hand side represents the frictional dissipation during the rearrangement of grains. The parameter μ is a function solely of the angle

of internal friction of the soil, ϕ.[1] The second term is the energy dissipated in fracture for an increase in surface dS of a volume V_s of solids which occupy a volume $V_s(1 + e)$. The parameter Γ is the critical strain energy release rate, a measure of the 'material toughness'.

Substituting

$$\mathrm{d}\bar{\varepsilon}^{\mathrm{p}} = -\frac{\mathrm{d}e^{\mathrm{p}}}{(1+e)} \tag{7}$$

we obtain

$$\mathrm{d}e^{\mathrm{p}} = \frac{\Gamma\,\mathrm{d}S}{(1-\mu)\bar{\sigma}V_s} \tag{8}$$

where μ is a function of ϕ alone.

Considering the new surface area as consisting of the area produced by each order of particle within the simulation gives

$$\mathrm{d}e \propto \frac{\Gamma}{(1-\mu)\bar{\sigma}} \sum_{r=0}^{r=s} B_r d_r \tag{9}$$

where the orders of the particle sizes are listed as $r = 0$ for the largest particle (size d_0) and $r = s$ for the smallest (size d_s), and B_r is the number of particles of size d_r which are splitting.

In calculating the value of de in the simulation, equation (9) was simplified to

$$\mathrm{d}e \propto \frac{K}{\bar{\sigma}} \sum_{r=0}^{r=s} B_r d_r \tag{10}$$

where K is taken to be a constant.

The uniformity coefficient U_c ($U_c = d_{60}/d_{10}$, where d_{60} is the particle size for which 60 per cent by mass of the particles are finer) is calculated and plotted after every increment of $\bar{\sigma}$. The particle-size distribution is plotted after every 35 increments of stress so that $\bar{\sigma}$ has increased by a factor of approximately $\sqrt{2}$.

Having calculated the survival probabilities, a pass is made through the triangles, selecting triangles for fracturing at random according to their survival probabilities. At the end of the pass, those triangles which have been selected are fractured and the survival probabilities are updated as necessary. A triangle is selected for fracture by generating a uniformly random number between 0 and 1, and selecting the triangle if this value is greater than its survival probability.

The intention was to repeat this process until no more particles split, before incrementing the stress. However, this was found to be too time consuming and the criterion was relaxed to repeating the process until less than 2 per cent of the total number were fractured. This was considered to given adequate accuracy.

Calculating and plotting the distribution of particle sizes, the void ratio and uniformity coefficient, using the above equations, is relatively straightforward. The main task then, as far as the program is concerned, is in deciding how to store and manipulate the triangles.

It is desirable that the representation of each triangle be concise because a large number of triangles is generated when there has been a significant increase in the stress. The more increments in stress which are permitted, the easier it is to observe trends in the void ratio and uniformity coefficient and the greater the number of particle-size distribution curves which can be plotted.

The size and co-ordination number required to calculate the survival probability should be easily extracted for any given triangle. When calculating the number of neighbours of a triangle, it is useful if these can be located without having to consider all other triangles in turn and without having to compare the coordinates of their vertices.

REPRESENTATION OF TRIANGLES

If the triangles are represented by their vertices, then the vertices of the two triangles resulting from a split can be easily calculated from the vertices of the triangle which has been fractured. Instead, for reasons of efficiency which are described later, each triangle is assigned a unique binary string, or sequence, and the algorithms to determine the adjacency of two arbitrary triangles or to locate the neighbours of a given triangle are based on the binary sequences rather than on the positions of the vertices. This also reduces the number of floating-point calculations required during the analysis.

The binary string for each new triangle produced is obtained by taking the string associated with the triangle being split and appending a '0' or '1' depending on whether the orientation of the new triangle corresponds to a $+135^\circ$ or -135° rotation (see Figure 2).

Figure 3 shows the binary strings of all triangles which have not been further sub-divided after 6 arbitrary splits have been made to a triangle with binary sequence '0'. Each digit also represents a reduction in size by a factor of two.

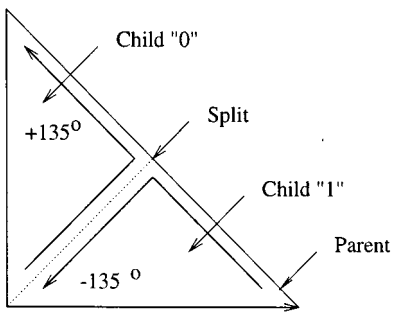

Figure 2. Assignment of additional digit to children of a triangle

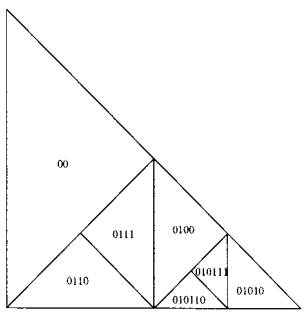

Figure 3. Binary strings after 6 arbitrary fractures

DERIVATION OF THE INITIAL GRID

The initial square grid of 50 triangles must be such that each triangle begins with a different string. This is achieved by considering an all-encompassing master triangle, with sequence '0'. This is then fractured a number of times with the minimum number of splits necessary to form the grid pattern. Triangles produced which lie outside the square boundary are not fractured further.

In general, for an initial grid which is i by j (containing $2ij$ triangles) this requires $2n$ passes through the existing triangles, where n is the smallest integer for which $i + j \leqslant 2^n$. At each pass, only those triangles which overlap or are within the grid are split. Figure 5 shows the pattern of triangles for the square grid used in the actual analysis and for a 6×10 grid, also indicating the triangles not belonging to the sample but which are in contact with it from above and from the right.

The triangles are stored in the code by one-dimensional arrays where the order of the elements corresponds to the order in which they were created. Four main arrays are used: $P[\,]$, $C0[\,]$, $C1[\,]$ and $Z[\,]$. The first three store information in chronological order; element 0 is the initial master triangle, element 1 is the next triangle to be formed. The P array stores the chronology of each element's parent, so that the statement $P[r] = m$ means that the parent of the rth element was the mth element. The $C0$ and $C1$ arrays similarly store the chronology of each child for each parent, so that $C0[m] = r$ means that one child of element m is element r, for example. $Z[\,]$ is used to

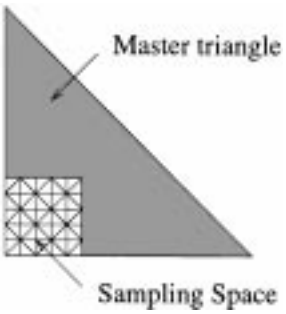

Figure 4. Master triangle and sampling space

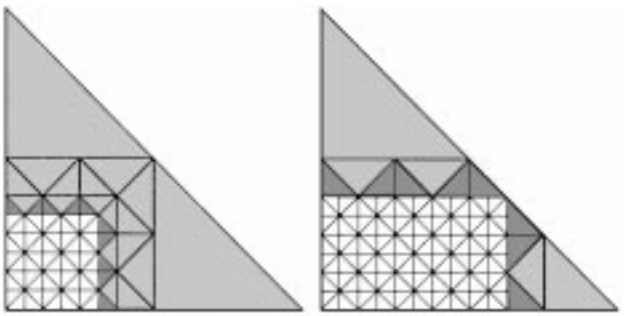

Figure 5. Triangles selected for 5×5 and 6×10 grids

store the last digit of the triangle's binary sequence, since all other digits can be extracted recursively using $P[\,]$ and $Z[\,]$.

In addition, an array of flags $F[\,]$ is used to indicate whether the triangles are split, within the sample, outside the sample but in contact with its top or right-hand side, or outside the sample and not in contact with its sides. For convenience, we represent the values of these flags as SPLIT, IN_SAMPLE, BOUNDARY, and DISCARDED.

The co-ordinates at the top right-hand corner of the grid are given by the x-co-ordinate $hi/2^n$ and the y-co-ordinate by $hj/2^n$ where h is the height of the master triangle, n is as before and, in this case, $i = j = 5$ for the sample space. Since the grid of 50 triangles is taken to have unit area, the value of h in the simulation is 3·2 and the area of the master triangle is 5·12. For each new triangle created during the $2n$ passes, the value of the flag stored in $F[\,]$ is determined by comparing these coordinates with the co-ordinates of the vertices of the triangle.

Since the memory required for each triangle is the same regardless of the length of the binary string, the storage required to produce a specified number of triangles can be determined in advance.

The number of element $N(k)$ required in the arrays in order to run the analysis until k triangles are present in the sample is given by $N(k) = 2(k + z) - 1$, where z is the number of un-fractured triangles not in the sample after the initial grid has been formed. For the square grid consisting of 50 triangles in Figure 5, z has a value of 28.

ADJACENCY AND CO-ORDINATION NUMBER

Let the sequences $\{a_i: i = 1, 2, \ldots, n\}$ and $\{b_i: i = 1, 2, \ldots, m\}$ represent the binary strings for two adjacent triangles, neither of which have been further sub-divided. If the first digit in which the sequences differ is the kth, then the sequence consisting of the first $(k - 1)$ digits represents the smallest triangle, formed during the splitting process, of which both triangles were once part: this is represented by triangle X in Figure 6. In addition, the following conditions must hold:

(a) From Figure 6, it follows that the $(k + 1)$th digit of each sequence, if it exists, must be the same as the kth.

$$\begin{aligned} a_{k+1} &= a_k, \quad k < n \\ b_{k+1} &= b_k, \quad k < m \end{aligned} \tag{11}$$

If these digits differ, for either triangle, then the triangles are descended from two triangles which are not adjacent and they therefore cannot be adjacent to each other.

(b) Every second digit following the $(k + 1)$th, for each sequence, must differ from the previous digit. This ensures that the triangles have sides which lie along (and on opposite sides of) the line segment formed by the fracture corresponding to the kth digit.

$$\begin{aligned} a_{k+1+2r} &\neq a_{k+2r}, \quad 0 < 2r < n - k \\ b_{k+1+2r} &\neq b_{k+2r}, \quad 0 < 2r < m - k \end{aligned} \tag{12}$$

(c) For the sides of the triangles which lie along this segment to overlap, the digits after the kth, where these exist for both sequences, must differ.

$$a_i \neq b_i, \quad k < i \leqslant \min\{n, m\} \tag{13}$$

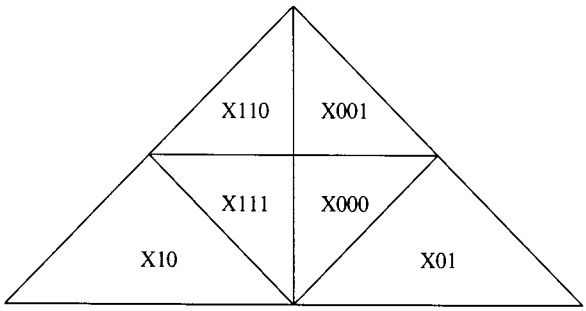

Figure 6. Splitting an arbitrary triangle 'X'

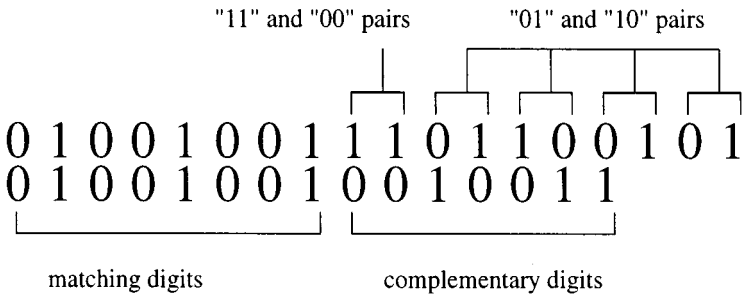

Figure 7. Comparing two adjacent triangles

This effectively means that the pattern of splits which correspond to these digits are mirrored in the line segment formed when the kth digits were produced.

If there is an odd number of digits following the $(k + 1)$th then this indicates that the side which is in contact with the other triangle is not the hypotenuse.

Figure 7 shows the comparison of the patterns for two adjacent triangles. The first triangle has an even number of digits following the '11' pair and is, therefore, in contact along its hypotenuse with the second triangle. Conversely, the second triangle is in contact along part of one of its two equal sides.

To determine the neighbours of any particular triangle we could consider all the other triangles in turn and check whether the above conditions apply. However, it is more efficient to perform a systematic search, described in the next section, using $P[]$, $C0[]$, $C1[]$ and $Z[]$. This also reduces the time required to calculate the co-ordination number when the number of triangle increases.

The co-ordination number of a triangle is the number of neighbours which share an edge contact and account must be taken of neighbours contacting the sample space. Contact with the top and right-hand side of the sample can be incorporated by checking the flags of $F[]$ and including those triangles which are outside the sample but in contact with its sides.

To determine whether the triangle is on the base or the left of the sample we can simply check whether the bit pattern of the binary string is '00' (for the left) or '01' (for the base), followed by

pairs of '01' and '10'. Since triangles cannot be both on the left and on the base, we can simply check for '01' and '10' patterns following the second digit.

$$a_{2r+1} \neq a_{2r+2}, \quad 0 < 2r < n-1 \tag{14}$$

The co-ordination numbers of the triangles resulting from a split will always be less than or equal to the co-ordination number of the triangle being split. Equality for both only occurs for a co-ordination number of 3.

SEARCHING FOR ADJACENT TRIANGLES

The algorithms used to search for triangles adjacent to a given triangle and thus calculate its co-ordination number, are described below. In brief, the search can be divided into three stages:

(1) Find the divergence points; those triangles for which the given triangle is descended from one child, and for which triangles adjacent of the given triangle might be descended from the other.
(2) Locate triangles adjacent to the given triangle which are either larger than or equal in size.
(3) Locate triangles adjacent to the given triangle which are smaller in size.

Stage 1

Let t be the element in the arrays which represents the given triangle. Since the arrays represent the chronology of the triangles, this triangle was also the tth triangle formed. Stage 1 considers the binary string triangle t and locates those digits corresponding to possible values of k, described in the previous section. It does this by considering the digits in reverse order as potential values of k and checking whether condition (12) is satisfied.

The last digit of the sequence is given by $Z[t]$. The rest of the digits are extracted from the $Z[\,]$ values of the triangle's ancestors; the ancestors being obtained successively using $P[\,]$. As the digits are extracted they are stored in an array, digit_buffer, whose length at each step is stored in the variable digit_count.

The last digit of the sequence of the given triangle could represent a possible value of k, i.e. the first digit which differs from the sequence of an adjacent triangle. Other possible candidates, using condition (11), are the first digits of pairs of consecutively equal digits, '00' or '11', occurring in the sequence.

However, if two pairs of consecutively equal digits are separated by an even number of digits, then the position of the first digit of the first pair cannot represent a valid value of k since the existence of second pair would violate condition (12). For example, in Figure 8, the seventh digit from the end is not a possible starting point because it is an even number of digits from the '00' pattern which follows. Hence there are, at most, three possible starting points for the search in stage 2.

The co-ordination number is initialized to zero at the start of Stage 1 and is increased as appropriate in the second and third stages. An array, pair_found, is used to store whether suitable values of k have been found at even or odd positions in the sequence (ignoring the digit at the end of the sequence). This avoids calling Stage 2 when condition (12) is know to be violated. At the end of the search, the current value of the pair_found is used to determine whether condition (14) is satisfied. If it is, then the given triangle lies on the base or the left of the sample space and the

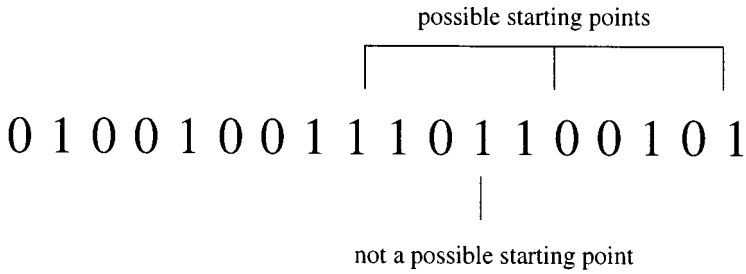

Figure 8. Possible starting points for stage 2 of the search

co-ordination number is increased by one. The algorithm for Stage 1 is as follows:

STEP 1: Let $i = t$, coordination = 0, pair_index = 0, digit_count = 0,
pair_found[0] = pair_found[1] = 0
STEP 2: Start forward search (Stage 2)
STEP 3: Let digit_buffer[digit_count] = $Z[i]$
Let digit_count = digit_count + 1
Let $i = P[i]$
STEP 4: If pair_found[pair_index] = 0 and $Z[i]$ = digit_buffer[digit_count − 1] then,
Start forward search (Stage 2)
Let pair_found[pair_index] = 1
STEP 5: Let pair_index = 1 − pair_index
STEP 6: If $P[i] \neq 0$
Repeat STEPS 2–5
STEP 7: If pair_found[pair_index] = 1
coordination = coordination + 1
(Triangle t was on the base or left of sample)

Stage 2

Stage 2 starts with the current values of i, digit_buffer and digit_count from State 1. The first step is to obtain the sibling of triangle i, and to then follow a path through its descendents, using $C0[\,]$ and $C1[\,]$, choosing the children so that condition (13) is satisfied. In order to do this, the children followed are those whose value of $Z[\,]$ complements the corresponding value is digit_buffer. For example, if the value stored in digit_buffer is a 0, the first child is obtained using $C1[\,]$, and so on. This continues until digit_buffer is exhausted. If, during this process, the value of $F[\,]$ for the current triangle indicates a value of IN_SAMPLE or BOUNDARY, then the current triangle is adjacent to the given triangle and so the co-ordination number is increased by one, and control returned to Stage 1. If the value of $F[\,]$ indicates a value of DISCARDED, control is returned to Stage 1 without the co-ordination numer being incremented.

If the end of the above process is reached without control being returned to Stage 1, then the current triangle is one which has been fractured. At this point we may be in a position to know what the next digit must be: If the value of digit_buffer was zero then, from condition (11), the last digit of the current triangle must be repeated. If the value of digit_buffer was even and non-zero

then, from condition (12), the next digit must complement the previous. For the either of these two cases the appropriate child is located and the previous check made on $F[\,]$. The reason for checking this extra digit is so that the search in Stage 3 may ignore all previous digits.

STEP 1: Let $j = P[i]$
if $Z[i] = 1$, then let $j = C0[j]$
otherwise, let $j = C1[j]$
STEP 2: count = digit_counter
(Initialize counter with the number of digits in digit_buffer)
STEP 3: if $F[j]$ = IN_SAMPLE or BOUNDARY
Let co-ordination = co-ordination + 1
Return from routine.
STEP 4: if $F[j]$ = DISCARDED then return from routine.
STEP 5.0: While counter $\neq 0$ perform STEPS 5·1–5·4
STEP 5.1: if digit_buffer[counter − 1] = 1, then let $j = C0[j]$
otherwise let $j = C1[j]$
STEP 5.2: if $F[j]$ = IN_SAMPLE or BOUNDARY
Let co-ordination = co-ordination + 1
Return from routine.
STEP 5.3: if $F[j]$ = DISCARDED then return from routine.
STEP 5.4: Let counter = counter − 1
STEP 6: if digit_counter = 0, then
if $Z[j] = 0$, let $j = C0[j]$
otherwise let $j = C1[j]$
STEP 7: if digit_counter is > 0, and even, then
if $Z[j] = 0$, let $j = C1[j]$
otherwise let $j = C0[j]$
STEP 8: if $F[j]$ = IN_SAMPLE or BOUNDARY, then
Let co-ordination = co-ordination + 1
Return from routine.
STEP 9: if $F[j]$ = DISCARDED then return from routine
STEP 10: Call Stage 3 with triangle j

Stage 3

This stage locates adjacent triangles which are smaller than the given triangle by following the children of the triangle passed in from Stage 2 which satisfy condition (12), It is not necessary to consider all the descendants of the triangle passed in from stage 2. Every second digit of the remaining sequence must be different from the previous digit, so we can group the digits in pairs and consider the possible combinations as illustrated in Figure 9, where the sequence '01' represent a branch to the left, and the sequence '10' represents a branch to the right.

To systematically search the branches we can adopted the rule that, starting from the left of the diagram, we attempt to take left branches wherever possible. Whenever we encounter a triangle which is un-split, we increment the co-ordination number and backtrack until we can take a right turn. After taking the right turn, we continue to try to take left turns, and so on. The search is complete when the starting position has been re-visited for the second time.

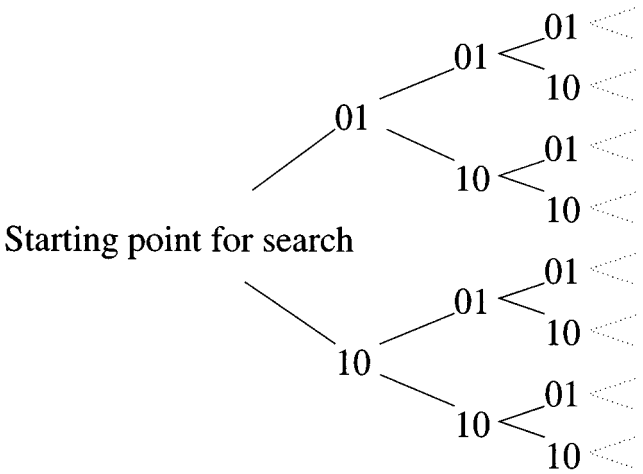

Figure 9. Considering the remaining digits as branches of a path

Ignoring the times when we are in the process of backtracking, we can consider being in four possible modes during the searching process:

mode 0: On a '0', an odd number of digits from the starting point. If split, then this is the start of a '01' pair.
mode 1: On a '1', an even number of digits from the starting point. This will be the second digit in a '01' pair.
mode 2: On a '1', an odd number of digits from the starting point. If split, then this is the start of a '10' pair.
mode 3: On a '0', an even number of digits from the starting point. This will be the second digit in a '10' pair.

Initially, we start in mode 0 and alternate between this mode and mode 1 while we are able to follow '01' pairs. When we encounter a triangle which has not been split we first increment the co-ordination number by one, and then backtrack either one digit (if in mode 0) or two (if in mode 1) to take us the last digit of the previous pair. We then take a 'right-turn', using $C1[\,]$, which takes us into mode 2. In taking the one or two steps back, we note whether this returns us to the starting point of the search. If it does, then we have completed the search of the upper set of branches in Figure 9 and the subsequent right turn takes us into the lower set.

From mode 2 we move to mode 3, if we can, to complete the '10' pair, and then return to mode 0. If we encounter an un-split triangle in either mode 2 or 3 we again increment the co-ordination number, backtrack one digit (if in mode 2) or two (if in mode 3), and then continue backtracking in groups of two digits at a time until we encounter a '1' or reach the starting point again. This point represents where we previously branched off to the right. If it is also the starting point passed in by Stage 2, then both the upper and lower branches in Figure 9 have been searched and the process is terminated, otherwise we take a further two steps back and use $C1[\,]$ to follow the branch with a '1' as the next digit, taking us back into mode 2. If this last step, of moving back two

digits and forward one, took us beyond the starting position, then we have moved from the upper set of branches in Figure 9 to the lower set. This leads to the following algorithm, where j is the triangle passed in from Stage 2.

STEP 1: Let $i = j$, visited_start = 0, mode = 0
STEP 2: If mode = 0 or 3
let $i = C0[i]$,
otherwise
let $i = C1[i]$,
STEP 3: Execute one of STEPS 4·0–4·3 depending on whether mode = 0, 1, 2, or 3, moving immediately to STEP 5 afterwards.
STEP 4.0: If $F[i] \neq$ SPLIT
let $i = P[i]$ and mode = 2
otherwise
let mode = 1
STEP 4.1: If $F[i] \neq$ SPLIT
let $i = P[P[i]]$ and mode = 2
otherwise
let mode = 0
STEP 4.2: If $F[i] \neq$ SPLIT
let $i = P[i]$
While $i >$ triangle and $Z[i] = 0$
let $i = P[P[i]]$
If $i >$ triangle
let $i = P[P[i]]$ and mode = 2
otherwise
let mode = 3
STEP 4.3: If $F[i] \neq$ SPLIT
let $i = PP[[i]]$
While $i >$ triangle and $Z[i] = 0$
let $i = P[P[i]]$
If $i >$ triangle
let $i = P[P[i]]$ and mode = 2
otherwise
let mode = 0
STEP 5: if $i <$ triangle
visited_home = visited_home + 1
STEP 6: if visited_home = 2
Return from routine.
STEP 7: Repeat STEPS 2–6

CALCULATING THE PROBABILITIES

When a triangle fracture, the survival probabilities and the co-ordination numbers of the new triangles need to be calculated. At most one other triangle can be affected by the split. This if it exists, must be a triangle which lies along the hypotenuse of the triangle being fractured, and

which is at least as large. Its co-ordination number will increase by one as the result of the fracture and its survival probability will need to be calculated again. It cannot be a triangle whose string first differs in the last digit of the triangle being fractured as this would mean that it lay on one of the equal sides of that triangle. Also, the digit in which the binary strings first differ must leave an odd number of digits remaining in the string of the triangle being fractured, in order for the other triangle to lie on its hypotenus. We therefore need only start at the end of the sequence of the given triangle and work towards the start, storing the values of each digit, and checking every other digit to see whether it is equal to the digit following it. If we reach such a point, then we take the parent of the current triangle and then move forwards again, following the digits which complement those which were stored. We need not check any further back because, even if any more pairs of equal digits were discovered, the pair found earlier would violate condition (12). When checking forward we need only check for as many digits remain in the string of the triangle being fractured as any adjacent triangle affected by the fracture cannot be smaller than the triangle being split.

The following routine finds the required triangle, if it exists, for a given triangle t, and sets triangle_affected to its position. A value of 0 is returned as a default. Since triangle 0 is known to be split, the program takes this to mean that no such triangle exists.

STEP 1: $i = t$, triangle_affected $= 0$, pair_index $= 1$, digit_count $= 0$
pair_found $= 0$
STEP 2: if pair_index $= 0$ and $Z[i]$ = digit_buffer[digit_count] then
pair_found $= 1$
STEP 3: digit_buffer[digit_count] $= Z[i]$
digit_count $=$ digit_count $+ 1$
pair_index $= 1 -$ pair_index
$i = P[i]$
STEP 4: if $i \neq 0$ and pair_found $= 0$ then
Repeat STEPS 2–3
STEP 5: if pair_found $= 0$
Return from routine
STEP 6: digit_count $=$ digit_count $- 1$
STEP 7: if digit_buffer[digit_count] $= 0$ then $i = C1[i]$
if digit_buffer[digit_count] $= 1$ then $i = C0[i]$
STEP 8: if $F[i]$ = IN_SAMPLE
triangle_affected $= i$
Return from routine
STEP 9: if $F[i] \neq$ SPLIT
Return from routine
STEP 10: if digit_count $= 0$
Return from routine
STEP 11: Repeat STEPS 6–10

ORIENTATION AND POSITION

There are eight possible orientations for the triangles produced. These can be numbered 0–7, according to the number of times the original triangle must be rotated by $+135^\circ$ to have the

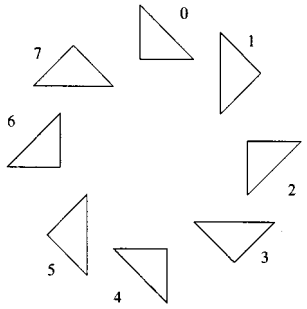

Figure 10. Possible orientations of triangles

same orientation (Figure 10). If the binary sequence for a triangle is $\{a_i: i = 1, 1, \ldots, n\}$, then the orientation can also be obtained by taking an initial value of 1, and adding or subtracting 1 for every '0' or '1' encountered in the sequence, adding or subtracting 8 each time the value falls below zero or becomes greater than 7 to keep it within the correct range.

The triangle and the triangles from which it was descended are represented by the subsequences formed by taking the first k terms of $\{a_i\}$. If the sequence of vector $\{M_n\}$ represent the vectors from the right-angle to the mid-point of the hypotenuse for each of these triangles, then each vector will be equal to the previous vector multiplied by a matrix which scales its magnitude by $\sqrt{2}$ and rotates it by $\pm 135^\circ$, so that

$$M_k = \left(\prod_{i=2}^{k} Q_i \right) M_1, \quad (1 < k \leqslant n)$$

where

$$Q_i = \begin{cases} \frac{1}{2} \begin{bmatrix} -1 & -1 \\ +1 & -1 \end{bmatrix}, & (a_i = 0) \\ \frac{1}{2} \begin{bmatrix} -1 & +1 \\ -1 & -1 \end{bmatrix}, & (a_i = 1) \end{cases}$$

The vertex at the right-angle of the triangle represented by $\{a_i\}$, is given by the sum of all the M_k except the last:

$$V_n = \sum_{k=1}^{n-1} \left(\prod_{i=2}^{k} Q_i \right) M_1$$

$$= Q_1 \cdots Q_{n-1} M_1 + Q_1 \cdots Q_{n-2} M_1 + \cdots + Q_1 Q_2 M_1 + Q_1 M_1 + M_1$$

This can be rewritten so that V_n is obtained from the recursive relation:

$$V_1 = M_1$$

$$V_k = Q_{n-k+1} V_{k-1}, \quad (k < 1 \leqslant n)$$

	Orientation							
	0	1	2	3	4	5	6	7
a	0	-1	1	-1	0	1	-1	1
b	1	-1	0	1	-1	1	0	-1
c	1	-1	0	1	-1	1	0	-1
d	0	1	-1	1	0	-1	1	-1

Figure 11. Values used to calculate positions of the 45° vertices

If we consider the value which is the smaller of the altitude and width of the triangle, then this value is halved on every second split. If this value is L_n for the triangle represented by a_i, then the vertices clockwise and anti-clockwise from the right angle are given by $(x + aL_n, y + bL_n)$ and $(x + cL_n, y + dL_n)$, where (x, y) is the position of the right angle and a, b, c, d have the values in Figure 11.

If the height of the master triangle is h then $M_1 = [^{h/2}_{h/2}]$ and the vertices of a triangle which occupies position i in the arrays can be calculated from $P[\,]$ and $Z[\,]$ as follows:

STEP 1: $L = h$, $x = 0$, $y = 0$, count 1, $t$type $= 1$

STEP 2: Calculate the next set of vertices using the recursive relationship and the appropriate value of Q_i

if $Z[i] = 0$

$x_$new $= (-x - y + h)/2$

$y_$new $= (x - y + h)/2$

t type $= t$type $+ 1$

if $Z[i] = 1$

$x_$new $= (-x + y + h)/2$

$y_$new $= (-x - y + h)/2$

t type $= t$type $- 1$

STEP 3: if $t$type < 0

$t$type $= t$type $+ 8$

if $t$type > 7

$t$type $= t$type $- 8$

STEP 4: if count $= 1$

$L = L/2$

STEP 5: count $= 1 -$ count

$x = x_$new

$y = y_$new

$i = P[i]$

STEP 6: if $i \neq 0$

Repeat STEPS 2–5

STEP 7: The vertices clockwise and anti-clockwise from the right angle are given by $(x + aL, y + bL)$ nd $(x + cL, y + dL)$, where a, b, c, d are obtained from the value of $t$type and the table in Figure 11.

DISCUSSION

The method of representing the triangles by binary strings and the algorithm to calculate the number of neighbours proved effective in supplying the speed of calculation required to produce a typical output plot (Figure 13), in which the simulation (using the values $m = 5$ and $a = 4$) was terminated after the number of triangles present in the sample was approximately 130,000. A plot of the fractured triangles is given in Figure 12.

Figure 13(a) shows the exponential shape of the developing curves of percentage by number of particles larger than d, versus d on a logarithmic scale. This is shown more clearly in Figure 13(b), when the percentage by numer of particles larger than d is also plotted on a logarithmic scale, the slope of the lines relating to a fractal dimension of approximately 1·36.[1] Figure 13(d) plots the cumulative mass fraction curve which shows that the grain size distribution curve in Figure 13(a) is tending towards a stable exponential curve, as the distribution of particles evolves. Figure 13(c) plots the logarithm of the absolute number of particles larger than d to the logarithm of d. For the chosen values of m and a, only the smallest particles are fracturing as the macroscopic stress $\bar{\sigma}$ is increased, since the high co-ordination values of the larger particles gives them high survival probabilities. Figure 13(e)shows the plot of voids ratio against logarithm of applied stress. As the model predicts the change in voids ratio rather than the absolute value, an arbitrary value of 0·7 was selected as the initial value. The curve has the expected shape, with less crushing evident at lower stresses, and the amount of breakage increasing as $\bar{\sigma}$ increases and fractal begin to form. Figure 13(f) shows that the uniformity coefficient increases to a constant value with increasing macroscopic stress. This is due to the evolution of a constant d_{60} size at lower stress levels [Figure 13(d)] and the tendency of the d_{10} size towards a constant value at higher stresses [Figure 13(g)]. McDowell *et al.*[1] describe the forms of these plots in terms of an emerging fractal distribution of particle sizes and find they compare favourably with experimental data.

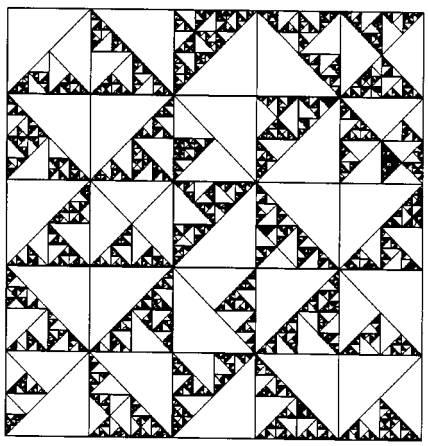

Figure 12. Pattern of fractured triangles at end of analysis

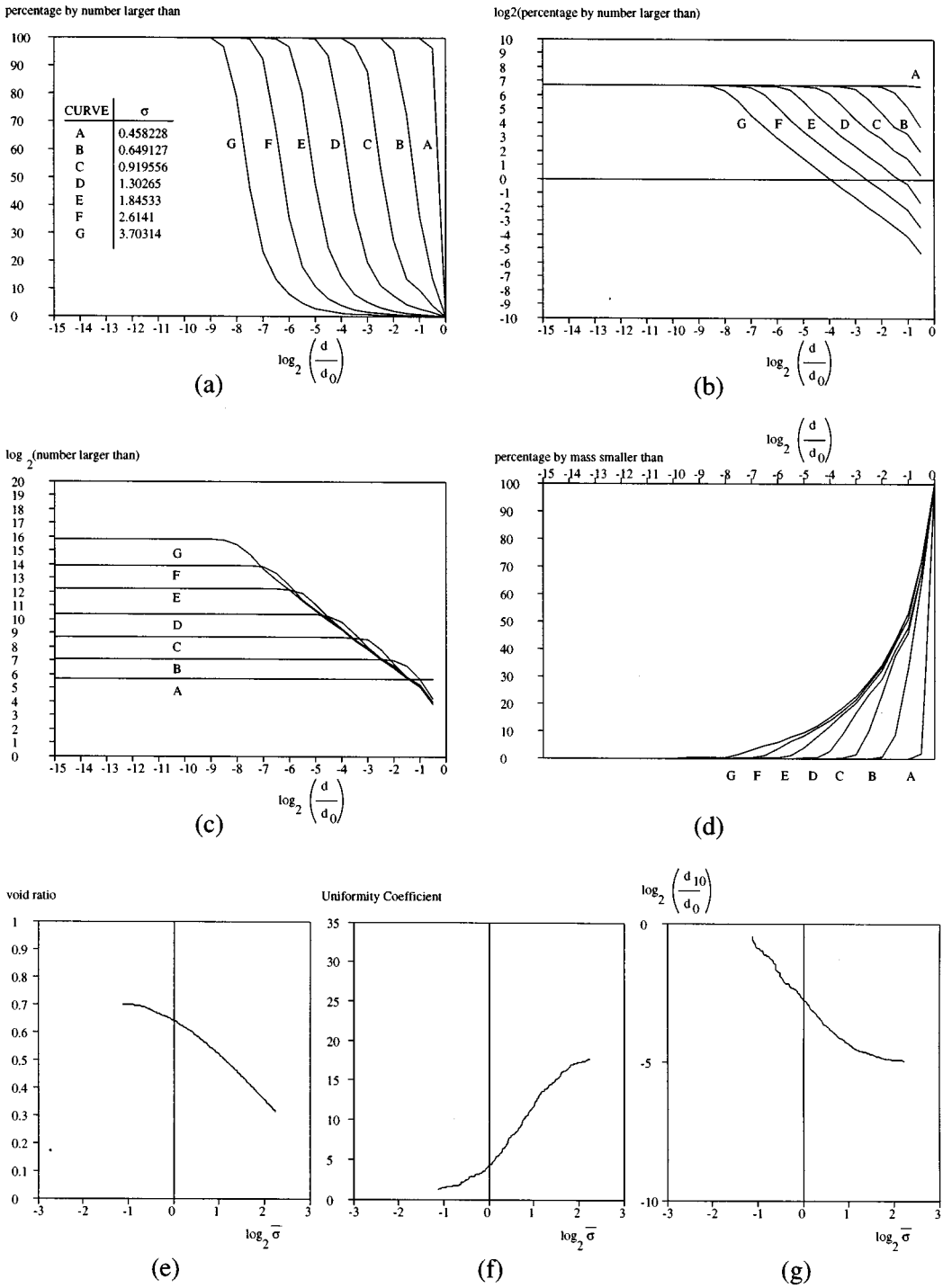

Figure 13. Example of output from the numerical simulation

In the present simulation the sequence boundary contributed a value of one to the co-ordination numbers of those particles whose edges lie along its sides. This would result in a tendency for those triangles to be more likely to break than similar sized triangles nearer the centre of the square. Instead, both a horizontal and vertical wrap-around could be incorporated so that triangles on the boundary contribute to the co-ordination number of triangles on the opposite sides.

Fractal distributions occur naturally in the debris formed by the collision of continental plates and in bodies of melange rock. It might be an interesting modification to allow for some of the original 50 triangles to have different properties, represented by different values of a and m in equation (1).

The model used by the simulation calculated the void ratio using a work equation. It might be useful to allow some of the initial triangles to explicitly represent voids. It is unlikely, however, that the above algorithms would be a suitable starting point to accommodate the possibility of sub-triangles falling or sliding into the voids as the fracture process proceeds.

CONCLUSIONS

1. An algorithm has been described which can track the size and co-ordination of grains within a sample as it fragments.

2 Weibull's statistical model of the strength of brittle ceramics has been applied to a soil/rock specimen in terms of the size, strength and co-ordination number of its constitutive grains.

3. This has been used to compute the surface energy created in a soil/rock sample as grain fracture occurs. If grain movement is neglected and voids, although not explicitly included in this model, are nevertheless inferred and presumed to compress, a work equation can be used to relate compression and applied stress. This offers an interesting avenue for the exploration of 'plastic' compression in granular materials.

ACKNOWLEDGEMENTS

Douglas Robertson thanks the EPSRC for the financial support of a student-ship. The authors are grateful to Hewlett–Packard for their donation of a computer, and for their stimulation of the CoLoS (Conceptual Learning of Science) initiative.

REFERENCES

1. G. R. McDowell, M. D. Bolton and D. Robertson, 'The fractal crushing of granular materils', *Int. J. Mech. Phys. Solids*, **44,** 2079–2102 (1996).
2. P. A. Cundall and O. D. L. Strack, 'A discrete numerical model for granular assemblies', *Geotechnique*, **29,** 47–65 (1979).
3. C. Sammis, G. King and R. Beigel, 'The kinematics of gouge deformation', *Pure Appl. Geophys.*, **125,** 777–812 (1987).
4. A. C. Palmer and T. J. O. Sanderson, 'Fractal crushing of ice and brittle solids', *Proc. R. Soc. Lond.*, *A* **433,** 469–477 (1991).
5. D. L. Turcotte, 'Fractals and fragmentation', *Geophys. Res.*, **91,** 1921–1926 (1986).
6. M. D. Bolton and G. R. McDowell, 'Clastic Mechanics', *IUTA—Symp. Mechanics of Granular and Porous Materials* (1986) (in press).
7. M. D. Lee, 'The angles of friction of granular fills', *Ph.D. Dissertation*, Cambridge University, 1992.
8. J. C. Jaeger, 'Failure of rocks under tensile conditions', *Int. J. Rock. Min. Sci.* **4,** 219–227 (1967).

